Jisy Raju,
Assistant Professor, CE Cherthala

Module 2

Relational Model: Structure of relational Databases, Integrity Constraints, synthesizing ER diagram to relational schema, Database Languages:Concept of DDL and DML relational algebra

2.1 Relational model in DBMS

In relational model, the data and relationships are represented by collection of inter-related tables. Each table is a group of column and rows, where column represents attribute of an entity and rows represents records. The table name and column names are helpful to interpret the meaning of values in each row. In the formal relational model terminology, a row is called a tuple, a column header is called an attribute, and the table is called a relation. The data type describing the types of values that can appear in each column is represented by a domain of possible values.

2.1.1 Structure of Relational model

Domains: A domain is a set of values permitted for an attribute in a table. Domain is atomic. For example, age can only be a positive integer. A data type or format is also specified for each domain. It is possible for several attributes to have the same domain. The data type for Employee_ages is an integer number. Some examples of domains follow:

- Mobile numbers, . The set of ten-digit phone numbers is valid
- Local_phone_number, The set of seven-digit phone numbers valid within a particular area code
- Social_security_numbers, . The set of valid nine-digit Social Security numbers.

Attribute: Each column in a Table. Attributes are the properties which define a relation. e.g., Student_Rollno, NAME,etc.

Tuple – It is nothing but a single row of a table, which contains a single record.

Relations- are in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.

Relation schema- A relational schema is the design for the table. It includes none of the actual data, but is like a blueprint or design for the table, so describes what columns are on the table and the data types. It may show basic table constraints (e.g. if a column can be null) but not how it relates to other tables.

A relation schema R, denoted by R (A1,A2, ..., An), is made up of a relation name R and a list of attributes, A1,A2, ...,An. Each attribute Ai is the name of a role played by some domain D in the relation schema R. D is called the domain of Ai and is denoted by dom(Ai). The relation schema R(A1,A2, ...,An), also denoted by r(R), is a set of n -tuples r= {t1,t2, ...,tm}.

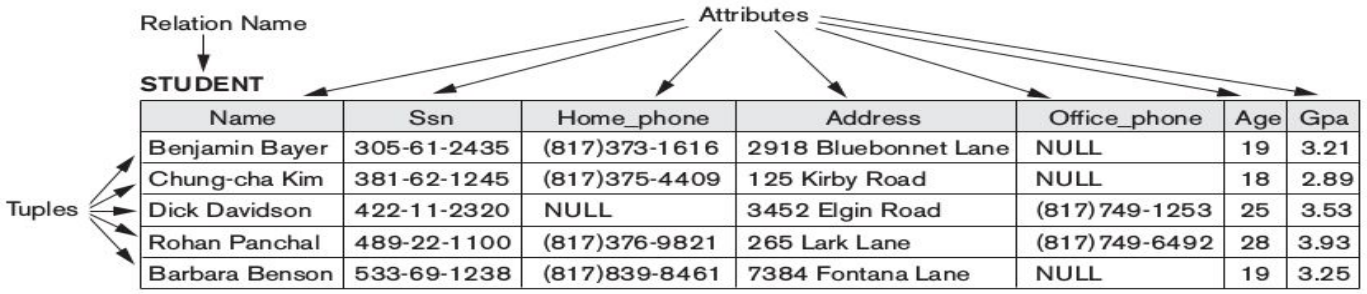
Degree- (or arity) of a relation is the number of attributes n of its relation schema. A relation of degree seven, which stores information about university students, would contain seven attributes describing each student. as follows:

STUDENT(Name,Ssn,Home_phone,Address,Office_phone,Age,Gpa)

Cardinality: Total number of rows present in the Table.

Relation instance – Relation instance is a finite set of tuples at a given time. The current relation state reflects only the valid tuples that represent a particular state of the real world.

Null value: A field with a NULL value is a field with no value. Primary key can't be a null value.



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Characteristics of Relations

- **Ordering of Tuples in a Relation:** A relation is defined as a set of tuples.The tuples in a relation do not have any particular order. In other words, a relation is not sensitive to the ordering of tuples. However, in a file, there always is an order among the records. This ordering indicates first, second, ith, and last records in the file. Tuple ordering is not part of a relation definition because a relation attempts to represent facts at a logical or abstract level. Many tuple orders can be specified on the same relation. For example, tuples in the STUDENT relation could be ordered by values of Name , Ssn , Age , or some other attribute. The definition of a relation does not specify any order: There is no preference for one ordering over another.

- **Ordering of attributes in a Relation:** The ordering of attributes is not important, because the attribute name appears with its value. There is no reason to prefer having one attribute value appear before another in a tuple. When a relation is implemented as a file, the attributes and the values within tuples are ordered.

- **Values in a tuple**: All values are considered atomic. A special null value is used to represent values that are unknown or inapplicable to certain tuples. In general, NULL values, means value unknown or value exists but is not available.

STUDENT	Name	SSN	HomePhone	Address	OfficePhone
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null

- **Interpretation (Meaning) of a Relation:** The relation schema can be interpreted as a declaration or a type of assertion. Each tuple in the relation can then be interpreted as a fact or a particular instance of the assertion. For example, the first tuple in Figure 3.1 asserts the fact that there is a STUDENT whose Name is Benjamin Bayer, Ssn is 305-61-2435, Age is 19, and so on.

Relational Model Notation

We will use the following notation in our presentation:

- A relation schema R of degree n is denoted by R(A 1 , A 2 , ..., A n).
- The uppercase letters Q, R, S denote relation names.
- The lowercase letters q, r, s denote relation states.
- The letters t, u, v denote tuples.
- In general, the name of a relation schema such as STUDENT also indicates the current set of tuples in that relation—the current relation state—whereas STUDENT (Name , Ssn , ...) refers only to the relation schema.
- An attribute A can be qualified with the relation name R to which it belongs by using the dot notation R.A—for example, STUDENT . Name or STUDENT . Age . This is because the same name may be used for two attributes in different relations. However, all attribute names in a particular relation must be distinct.
- An n-tuple t in a relation r(R) is denoted by t = <v 1 , v 2 , ..., v n >, where v i is the value corresponding to attribute A i . The following notation refers to component values of tuples:
- Both t[A i] and t.A i (and sometimes t[i]) refer to the value v i in t for attribute A i .

As an example, consider the tuple t = <‘Barbara Benson’, ‘533-69-1238’, ‘(817)839-8461’, ‘7384 Fontana Lane’, NULL , 19, 3.25> from the STUDENT relation in Figure 3.1; we have t[Name] = <‘Barbara Benson’>, and t[Ssn , Gpa , Age] = <‘533-69-1238’.

2.2 Relational Model Constraints

Constraints enforce limits to the data or restriction of data that can be inserted/updated/deleted from a table. The whole purpose of constraints is to maintain the data integrity during an update/delete/insert into a table. Constraints on databases can generally be divided into three main categories:

1. Constraints that are inherent in the data model. We call these inherent model-based constraints or implicit constraints.

2. Constraints that can be directly expressed in schemas of the data model, typically by specifying them in the DDL.

3. Constraints that cannot be directly expressed in the schemas of the data model, and hence must be expressed and enforced by the application programs. This is known as application-based or semantic constraints or business rules.

The schema-based constraints include domain constraints, key constraints, constraints on NULL s, entity integrity constraints, and referential integrity constraints.

Domain constraints:
Each table has certain set of columns and each column allows a same type of data, based on its data type. The column does not accept values of any other data type.

Domain constraints are **user defined data type** and we can define them like this:

Domain Constraint = data type + Constraints (NOT NULL / UNIQUE / PRIMARY KEY / FOREIGN KEY / CHECK / DEFAULT)

STU_ID	Name	Age
S001	Akshay	20
S002	Abhishek	21
S003	Shashank	20
S004	Rahul	A

Here, value **'A'** is not allowed since only integer values can be taken by the age attribute.

Key constraints

An attribute that can uniquely identify a tuple in a relation is called the key of the table. All the values of primary key must be unique.

STU_ID	Name	Age
S001	Akshay	20
S001	Abhishek	21
S003	Shashank	20
S004	Rahul	20

This relation does not satisfy the key constraint as here all the values of primary key are not unique.

Entity Integrity Constraint

Entity integrity constraint specifies that no attribute of primary key must contain a null value in any relation. This is because the presence of null value in the primary key violates the uniqueness property.

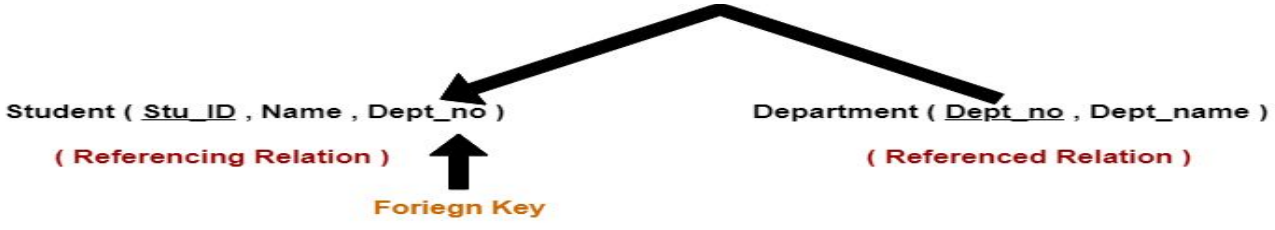
STU_ID	Name	Age
S001	Akshay	20
S002	Abhishek	21
S003	Shashank	20
	Rahul	20

This relation does not satisfy the entity integrity constraint as here the primary key contains a NULL value.

Referential Integrity Constraint-

Referential integrity constraints work on the concept of Foreign Keys. A foreign key is a key attribute of a relation that can be referred in other relation.

A set of attributes FK in relation schema R 1 is a foreign key of R 1 that references relation R 2 if it satisfies. R 1 is called the referencing relation and R 2 is the referenced relation.



Student

STU_ID	Name	Dept_no
S001	Akshay	D10
S002	Abhishek	D10
S003	Shashank	D11
S004	Rahul	D14

Department

Dept_no	Dept_name
D10	ASET
D11	ALS
D12	ASFL
D13	ASHS

Here,

- The relation ‘Student’ does not satisfy the referential integrity constraint.
- This is because in relation ‘Department’, no value of primary key specifies department no. 14.
- Thus, referential integrity constraint is violated.

A trigger is a *statement* or a *block of statement* which are executed automatically by the system when an event like insert, update or delete takes place on a table.Triggers are more powerful because the can check conditions and also modify the data.

Assertions - An assertion is a piece of SQL which makes sure a condition is satisfied or it stops action being taken on a database object. It could mean locking out the whole table or even the whole database. Assertions do not modify the data, they only check certain conditions.

Operations in Relational Model with Constraint Violations

Four basic operations performed on relational database model are Insert, update, delete and select.

- **Insert Operation:** Insert is used to insert data into the relation Insert can violate any of the **four types** of constraints. Domain constraints can be violated if an attribute value is given that does not appear in the corresponding domain or is not of the appropriate data type. Key constraints can be violated if a key value in the new tuple t already exists in another tuple in the relation r(R). Entity integrity can be violated if any part of the primary key of the new tuple t is NULL . Referential integrity can be violated if the value of any foreign key in t refers to a tuple that does not exist in the referenced relation. If an insertion violates one or more constraints, the default option is to reject the insertion. If the insertion is not rejected then, the insertion violation can cause cascade in the relation. A foreign key with **cascade** delete means that if a record in the parent table is deleted, then the corresponding records in the child table will automatically be deleted. This is called a **cascade** delete.

- **Delete Operation:** is used to delete tuples from the table. The Delete operation can violate only **referential integrity**. This occurs if the tuple being deleted is referenced by foreign keys from other tuples in the database. Here are some examples.
  -Operation: Delete the Department tuple with Dept NO = 1.
  Result: This deletion is acceptable and deletes exactly one tuple.
  -Operation:Delete the Student tuple with Dept No = 1.

Result: This deletion is not acceptable, because there are tuples in Department that refer to this tuple.

Several options are available if a deletion operation causes a violation. The first option, called restrict, is to reject the deletion. The second option, called cascade. A third option, called set null or set default, is to modify the referencing attribute values that cause the violation.And also combinations of these three options are also possible.

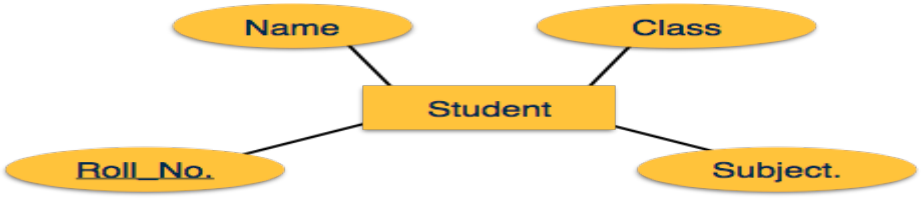
- **Update Operation:**Modify allows you to change the values of some attributes in existing tuples. Consider two table EMPLOYEE(Ssn, name, salary, Dno) and DEPARTMENT(Dno, Dname)
  -Operation:Update the salary of the EMPLOYEE tuple with Ssn = '123' to 2800.
  Result: Acceptable.
  -Operation:Update the Dno of the EMPLOYEE tuple with Ssn = '123' to 7.
  Result: Unacceptable, because it violates referential integrity.
  -Operation:Update the Ssn of the EMPLOYEE tuple with Ssn = '123' to '321'.
  Result: Unacceptable, because it violates primary key constraint

Updating an attribute that is neither part of a primary key nor of a foreign key usually causes no problems.

- **The Transaction Concept:** A transaction is an executing program that includes some database operations, such as reading from the database, or applying insertions, deletions, or updates to the database. At the end of the transaction, it must leave the database in a valid or consistent state that satisfies all the constraints specified on the database schema. A single transaction may involve any number of retrieval operations C and any number of update operations. For example, a transaction to apply a bank withdrawal will typically read the user account record, check if there is a sufficient balance, and then update the record by the withdrawal amount.

2.3 Synthesizing ER diagram to relational schema

An entity type within ER diagram is turned into a table. Each attribute turns into a column (attribute) in the table. The key attribute of the entity is the primary key of the table which is usually underlined. Derived attributes are ignored and multivalued attributes are represented in another table. Taking the following simple ER diagram of **strong entity,**

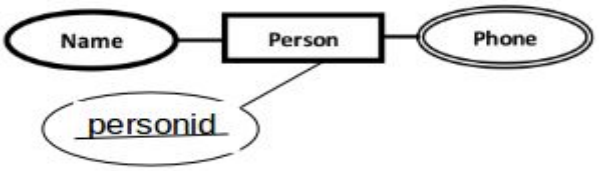


The initial relational schema is expressed in the following format writing the table names with the attributes list inside a parentheses as shown below for

Student(Roll_No , Name, Class, Subject)

- Create table for each entity.
- Entity's attributes should becomes column of tables with their respective data types.
- Declare primary key.

A **multi-valued attribute** is usually represented with a double-line oval.



If you have a multi-valued attribute, take the attribute and turn it into a new entity or table of its own.

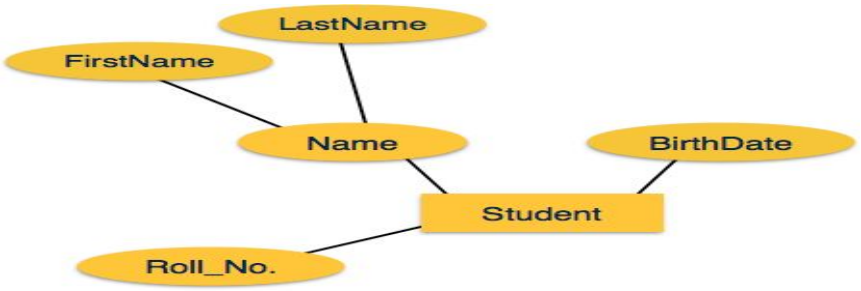
Person(personid , name)

Phones (***personid***, phone)

Phones
PhoneID(pk) Personid (fk) phone

Person
Personid (pk) name

A **composite attribute**,



The firstname and lastname become a individual attributes in a relational tables

Student(Roll_No, BirthDate, FirstName,LastName)

Mapping Relationship

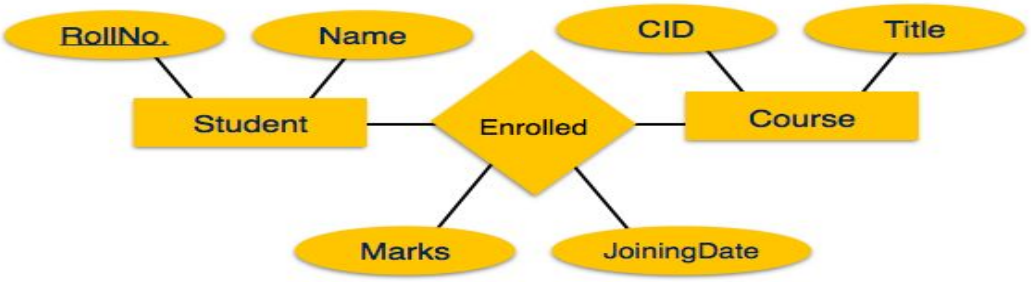
A relationship is an association among entities.

-One to many

-many to many

-many to one

Many to Many

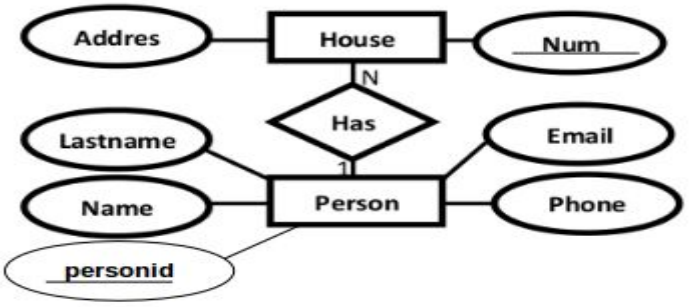


- Create table for each entity.
- Entity's attributes should becomes column of tables with their respective data types.
- Declare primary key.
- Create table for a relationship.
- Add the primary keys of all participating Entities as column of table with their respective data types.
- If relationship has any attribute, add each attribute as column of table.

- Declare a primary key composing all the primary keys of participating entities.
- Declare all foreign key constraints

Student(RollNo,Name)
Course(CID, Title)
Enrolled(RollNo, CID, marks,JoiningDates)

One to many and Many to One



- Create table for each entity.
- Entity's attributes should becomes column of tables with their respective data types.
- Set primary Key of a table as a Foreign key of other table

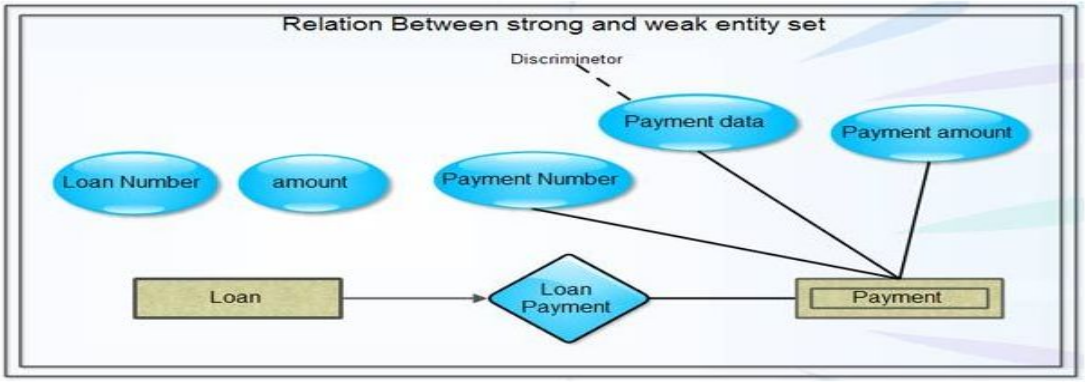
House(Num,Address, personid)
Person(personid,Name,Lastname,Email,Phone)

Or

House(Num,Address)
Person(personid,Name,Lastname,Email,Phone,Num)

Mapping Weak Entity Sets

A weak entity set is one which does not have any primary key associated with it.



- Create table for weak entity set.
- Add all its attributes to table as field.
- Add the primary key of linked strong entity set.
- Declare all foreign key constraints

Loan(Loan Number, amount,Payment Number)
Payment(Loan Number,Payment Number, Payment Data, Payment amount)

2.4 Database Languages:Concept of DDL and DML relational algebra

Relational algebra is a widely used procedural query language. It collects instances of relations as input and gives occurrences of relations as output. It uses various operation to perform this action.

Basic Relational Algebra Operations: Relational Algebra divided in various groups

Unary Relational Operations

- SELECT (symbol: σ)
- PROJECT (symbol: π)
- RENAME (symbol:ρ)

Relational Algebra Operations From Set Theory

- UNION (υ)
- INTERSECTION ( ),
- DIFFERENCE (-)

Binary Relational Operations

- JOIN
- DIVISION
- CARTESIAN PRODUCT (x)

SELECT (σ): The SELECT operation is used for selecting a subset of the tuples according to a given selection condition. Sigma(σ)Symbol denotes it.

$$\sigma_{\text{condition}} (\text{Relation})$$

STUDENT

ID	NAME	MARK
1	Jisy	70
2	vishnu	75
3	Dwayne	80

1. To retrieve entire details from STUDENT.
   σ (STUDENT)
2. To retrieve details from STUDENT where ID=2
   $\sigma_{\text{ID=2}}$ (STUDENT)

Projection(π): The project operation is used for selecting attributes according to a given selection condition.

$$\Pi_{\text{Attribute}} (\text{Relation})$$

1. To retrieve ID and NAME from STUDENT.
   $\Pi_{\text{ID, NAME}}$ (STUDENT)

Combination of Select and Project: Retrieve ID and Name from STUDENT where mark>=75

$\Pi_{\text{ID, NAME}} (\sigma_{\text{Mark>=75}}$ (STUDENT))

Rename Operation (ρ):

. The rename operation allows us to rename the oldt relation to new. ‘Rename’ operation is denoted with small Greek letter rho ρ.
ρ New (Old Relation Name)

Union operation (υ):UNION is symbolized by ∪ symbol. It includes all tuples that are in tables A or in B. It also eliminates duplicate tuples. A ∪ B

For a union operation to be valid, the following conditions must hold -

- R and S must be the same number of attributes.
- Attribute domains need to be compatible.
- Duplicate tuples should be automatically removed.

Retrieve students name **either** participant in arts **or** sports:

Π NAME (ARTS) ∪ Π NAME (SPORTS)

ARTS

ID	NAME
1	A
2	B
3	C

SPORTS

ID	NAME
3	C
2	B
4	D

ARTS **U** SPORTS

ID	NAME
1	A
2	B
3	C
4	D

Intersection : *A* ∩ *B* of two sets *A* and *B* is the set that contains all elements of *A* that also belong to *B* (or equivalently, all elements of *B* that also belong to *A*), but no other elements.

Retrieve students name those who participant in **both** arts **and** sports:

Π NAME (ARTS) ∩ Π NAME (SPORTS)

ARTS ∩ SPORTS

ID	NAME
2	B
3	C

Set Difference (−): The result of set difference operation is tuples, which are present in one relation but are not in the second relation.
Retrieve students name those who participant **only** in arts **and not** in sports:

Π NAME (ARTS) − Π NAME (SPORTS)

ARTS − SPORTS

ID	NAME
1	A

Retrieve students name those who participant **only** in sports **and not** in arts:

Π NAME (SPORTS) − Π NAME (ARTS)

SPORTS − ARTS

ID	NAME
4	D

Cartesian Product (X): Combines information of two different relations into one**.**

A

K	Y
1	A
2	B

B

K	Y
1	C
3	D

A **X** B

K	Y	K	Y
1	A	1	C
1	A	3	D
2	B	1	C
2	B	3	D

Join operation (⋈): Used to combine two tables having same attributes.Two type,

- Inner join
- Outer Join

Inner join ⋈**:**If the join criterion is based on equality of column values, the result is called an equijoin. A natural join is an equijoin with redundant columns removed

STUD

ID	NAME	DNO.
1	A	1
2	B	2
3	C	3

DEPT

DNO	DNAME
1	CS
2	EC
4	EE

STUD ⋈ DEPT

ID	NAME	DNO.	DNAME
1	A	1	CS
2	B	2	EC

Outer joins: Includes all the tuples from the participating relations in the resulting relation. There are three kinds of outer joins:

- Left outer join
- Right outer join
- Full outer join.

Left Outer Join(R ⟕ S):All the tuples from the Left relation, R, are included in the resulting relation. If there are tuples in R without any matching tuple in the Right relation S, then the S-attributes of the resulting relation are made NULL.

STUD ⟕ DEPT

ID	NAME	DNO.	DNAME
1	A	1	CS
2	B	2	EC
3	C	3	Null

Right Outer Join: (R ⟖ S): All the tuples from the Right relation, S, are included in the resulting relation. If there are tuples in S without any matching tuple in R, then the R-attributes of resulting relation are made NULL.

STUD⟖ DEPT

ID	NAME	DNO.	DNAME
1	A	1	CS
2	B	2	EC
Null	Null	4	EE

Full Outer Join: (R ⟗ S): All the tuples from both participating relations are included in the resulting relation. If there are no matching tuples for both relations, their respective unmatched attributes are made NULL.

STUD ⟗ DEPT

ID	NAME	DNO.	DNAME
1	A	1	CS
2	B	2	EC
3	C	3	Null
Null	Null	4	EE

Division operation (÷): The Division operator results in columns values in one table for which there are other matching column values corresponding to every row in another table.

A

K	X	Y
10	1101	A
10	1201	B
10	1301	C
20	1201	B
30	1101	A
30	1201	B
30	1301	C

B (divisor)

X	Y
1101	A
1201	B
1301	C

Result

K
10
30